Testing Documentation

Arana Charlebois, Tristan
Carvalho, Cam
Corbett, Cole
Kassie, Turner
Pauls, Andrew
Wallace, Jordan

Cosc 4P02
Dr. Naser Ezzati-Jivan

April 27, 2025

Contents

Preface

For the complete final report for our Cosc 4P02 project, please refer to the file named *Cosc-4p02-Final-Report.pdf* within the following repository `https://github.com/BrockU-4P02-Logistics-System/Frontend/tree/main/Documentation`. The following document is designed to represent a section of the overall final report.

1 Introduction

The following document is to present the documentation of the tests for our logistics system, to ensure that it exhibits proper functionality for users. This document is being presented in two main sections, core functionality testing, along with additional tests including stress testing, user timing tests, and unit testing. All tests regarding user interactions are performed and documented in the same manner, ensuring consistency. Further, backend testing and database testing deploys the use of unit (via *jest* and *junit*) tests. We link these tests so the reader may observe the code used for the testing of these respective component.

2 Core Functionality Testing

The following subsections include core functionality tests, defined for the initial MVP. All tests follow the same format, being test number, name, input, steps, conditions and expected/observed output. These tests served as the initial check to ensure that the systems main functions were exhibiting the desired functionality. These tests cases were created in a semi-weekly meeting by members as the first check for the system.

2.1 Routing

Title	Routing 001 - 50 Stops, Across North America
Input Data	50 Stop route
Test Steps	1. Navigate to dashboard while signed in with credits. 2. Fill in route of 50 stops across NA (Include Cabo, Toronto and begin in Kansas City). 3. Set number of drivers to one.
Expected Result	This should return a valid route of travel to the user.
Preconditions	Valid account with credits.
Postconditions	N/A.
Result	**Timeout**.

Title	Routing 002 - 50 Stops, Same Region
Input Data	50 Stop route within southern Ontario.
Test Steps	1. Navigate to dashboard while signed in with credits. 2. Fill in route of 50 stops across southern Ontario.
Expected Result	This should return a valid route of travel to the user, provided that the server does not timeout.
Preconditions	Valid account with credits.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	Routing 003 - Apply Route Options
Input Data	Random route involving separate cities.
Test Steps	1. Navigate to dashboard while signed in with credits. 2. Fill in route and ensure that "return to home" and "avoid highways" are selected in route options. 3. Ensure that the routes will use a highway (used Toronto to Niagara Falls).
Expected Result	This should return a route that does indeed return home and avoids highways.
Preconditions	Valid account with credits.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	Routing 004 - Unreachable Location By Auto
Input Data	Random route involving location in Greenland.
Test Steps	1. Navigate to dashboard while signed in with credits. 2. Fill in route and ensure that one of the locations is in Greenland, while another location is in Canada.
Expected Result	This should return a route that exhibits functionality of a plane when needed and notify the user that one of the stops is unreachable.
Preconditions	Valid account with credits.
Postconditions	N/A.
Result	Pass with desired functionality.

2.2 User Account

Title	User Account 001 - Invalid Password
Input Data	testpassword123!
Test Steps	1. Navigate to the sign-up page. 2. Fill in all other fields with valid entries. 3. Password contains invalid input.
Expected Result	This should deny the account creation, as the password doesn't contain an uppercase letter. This should be communicated to the user.
Preconditions	All other fields are valid, email has not been used.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	User Account 002 - Repeating Email
Input Data	tk21rx@brocku.ca
Test Steps	1. Navigate to the sign-up page. 2. Fill in all other fields with valid entries. 3. Enter repeated email.
Expected Result	This should deny the account creation, as this email already corresponds to an account and communicate this to the user.
Preconditions	All other fields are valid and tk21rx@brocku.ca is already registered.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	User Account 003 - Open Sessions, Same System
Input Data	Two sessions on the same computer
Test Steps	1. Sign in to user account. 2. Open another tab and view the functionality of the system.
Expected Result	This should allow the user directly into the application without needing to sign in.
Preconditions	User is logged into the system on the same device.
Postconditions	User remains logged in on the same device.
Result	Pass with desired functionality.

Title	User Account 004 - Resetting Password
Input Data	tk21rx@brocku.ca
Test Steps	1. On the login page, follow the ”Forgot Password” link. 2. Follow the link sent to user email to reset password. 3. Resign in to the system with new password.
Expected Result	Within the inbox of the email specified in the recovery field, users should receive an email, and be able to reset their password
Preconditions	tk21rx@brocku.ca is already registered as a valid email corresponding to an account.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	User Account 005 - Login With Google
Input Data	Valid Google account
Test Steps	1. On the login page, follow the "Sign in with Google" button. 2. Input a valid password and username to complete sign-in.
Expected Result	Proper profile creation using the valid Google account.
Preconditions	Registering with a valid Google account, not already tied to user account.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	User Account 006 - Account Creation with Email Already Registered with Sign in with Google
Input Data	Gmail used in registration field, used in previous test (User Account 005).
Test Steps	1. Navigate to the sign up page. 2. Sign up for a new account using the Google account used in Test 005.
Expected Result	Deny the account since the email has already been used.
Preconditions	Registering with a valid Google account, used for another account already.
Postconditions	N/A.
Result	Pass with desired functionality.

2.3 Credit System

Title	Credits 001 - Single Route Cost
Input Data	Form and calculate a single-TSP route and submit to the system.
Test Steps	1. Before submission, note the number of credits saved to the account. 2. Form a general, valid single driver route. 3. Calculate route, noting number of credits.
Expected Result	Test account currently holds 950 credits, so after calculation, this should be 940.
Preconditions	Testing with valid account, credits in account and valid route.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	Credits 002 - Multi-TSP Route
Input Data	Form and calculate a multi-TSP route and submit to the system.
Test Steps	1. Before submission, note the number of credits saved to the account. 2. Form a general, multi driver route (using two drivers for this test). 3. Calculate route, noting credits in account.
Expected Result	Test account currently holds 940 credits, so after calculation, this should be 920.
Preconditions	Testing with valid account, credits in account and valid route.
Postconditions	Valid credit deduction for multi-TSP.
Result	Pass with desired functionality.

Title	Credits 003 - Route Saving Costs
Input Data	Form a route and save to the system.
Test Steps	1. Compute a valid route for travel, single driver. 2. Save route to the system and note the overall credits on the account.
Expected Result	Test account currently holds 920 credits, so after saving, this should be 900 (calculating route and saving).
Preconditions	Fewer than six routes in the user's account, valid account, credits in account.
Postconditions	Total deduction of 20 credits.
Result	Pass with desired functionality.

2.4 Saving/Loading Routes

Title	Saving/Reloading 001 - Saving Route
Input Data	Valid computed route to store.
Test Steps	1. Form a general single driver route. 2. Calculate route and save a route with sufficient credits. 3. Navigate to load page to verify saved route.
Expected Result	Route should be stored in the database.
Preconditions	Testing with valid account, less than 6 routes already stored, sufficient credits.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	Saving/Reloading 002 - Maximum Saved Routes
Input Data	Saving a valid route to an account with six routes previously stored.
Test Steps	1. Form and compute a valid route for travel. 2. Ensure that account has 6 routes already saved. 3. Attempt to store more than 6 routes in an account.
Expected Result	Action should be denied. Each user can only have 6 routes. A message should notify the user.
Preconditions	Valid account with 6 saved routes.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	Saving/Reloading 003 - Storing Boolean Flags
Input Data	Saving a valid route to account with all boolean flags selected.
Test Steps	1. Calculate a route avoiding tolls, highways, ferries, and returning home. 2. Once route has been calculated, store route to account. 3. From the load page, reload the route.
Expected Result	Route should save and reload with the same flags applied.
Preconditions	All route flags selected when calculating, account has credits for calculating and saving.
Postconditions	N/A.
Result	Pass with desired functionality.

Title	Saving/Reloading 004 - Storing Boolean Flags Then Altering Route
Input Data	Saving a valid route to account with all boolean flags selected.
Test Steps	1. Calculate a route avoiding tolls, highways, ferries, and returning home. 2. Save this route to account. 3. Reload the route, then alter the flags, without resaving. 4. Reload the account from the account once again.
Expected Result	Route should save and reload with the same flags applied, even after modifying the route in between ensure that the route in DB isn't altered.
Preconditions	All route flags selected, saving to an account with credits.
Postconditions	N/A.
Result	Pass with desired functionality.

3 Additional Tests

We now give some additional tests for the system, particularly to answer questions involving timing from the user side of the system.

3.0.1 Determining the Largest Route

The following tests were all conducted in the following manner.

- On dashboard form routes.
- Note aspects pertaining to number of drivers, area of stops.
- Time the process of optimizing route, and then receiving the information on the frontend of the system, from a user perspective.

We use these tests for the main areas, namely, timing of the system from a user perspective and determining the largest routes that can be completed based on current timeouts placed on both the backend of the system and Vercel. These results are presented in table 1.

3.1 Testing Multiple Routes Simultaneously

Next, we present the following tests, namely, for determining the scalability of the system. The following tests were all conducted in the same manner, with each one further scaling up the number of requests made to the system by making use of multiple tabs making requests.

Stops	**Drivers**	**Route/Region**	**Result**	**Timing**
25	5	Niagara	Success	5 secs
25	5	North America	Failure	Timeout
2	1	Niagara to Toronto	Success	1 sec
5	1	Niagara to Toronto	Success	2 secs
10	1	Niagara to Toronto	Success	4 secs
20	1	Niagara to Toronto	Success	5 secs
40	1	Niagara to Toronto	Success	16 secs
50	1	Niagara to Toronto	Success	26 secs
55	1	Niagara to Toronto	Success	30 secs
57	1	Niagara to Toronto	Success	31 secs
58	1	Niagara to Toronto	Success	31 secs
59	1	Niagara to Toronto	Failure	Timeout
60	1	Niagara to Toronto	Failure	Timeout

Table 1: Test Results Routes

- Two times are recorded for each test.
- The first time indicates when all marker get their new, ordered numbers.
- Second time indicates when the map has been completely rendered.

Test 1: Single Request	
Description	Just a regular 22 stop route, 5 drivers.
Corresponding time	Time of a 22 stop route, across NA, 1 driver
Marker Assignment	9.5 seconds
Complete Render	39 seconds

Test 2: 2 Simultaneous Requests	
Description	Two tabs open, two 22 stop routes, 5 drivers each. First tab starts by hitting optimize route, second is timed when hitting optimize route.
Corresponding time	Time of a 22 stop route: 5 drivers (second tab)
Marker Assignment	10.5 seconds
Complete Render	42 seconds

4 Unit Testing

Unit testing was conducted on both the GA and database for ensuring proper functionality. The code for the tests conducted is referenced in their respective sections.

Test 3: 4 Simultaneous Requests	
Description	Four tabs, each with different 22 stop routes, 5 drivers. Timer starts after clicking optimize route on the 4th tab.
Corresponding time	Time of a 22 stop route: 5 drivers (fourth tab)
Marker Assignment	9 seconds
Complete Render	37 seconds

Test 4: 8 Simultaneous Requests	
Description	8 tabs open, each with 22 stop routes and 5 drivers. Tabs hit optimize route in quick succession. Timer starts on the 8th submitted calculate route.
Corresponding time	Time of a 22 stop route: 5 drivers (eighth tab)
Marker Assignment	9.5 seconds
Complete Render	39 seconds

Test 5: 9 Simultaneous Requests	
Description	9 tabs open, each with their own 22 stop unique route and 4–5 drivers. Tabs triggered in quick succession.
Corresponding time	Intended to test for slowdown with 9 vs. 8 simultaneous requests (ninth tab).
Marker Assignment	8.5 seconds
Complete Render	37 seconds

4.0.1 GA Unit Testing

For unit testing regarding the GA, please follow https://github.com/BrockU-4P02-Logistics-System/Backend/tree/master/src/test/java/org/example. These tests were concerned with testing core GA functions, namely, crossover, mutation and crossover.

4.0.2 Database Testing

For testing regarding the database, we have *jest* for inserting and retrieving information from the database as well as authorization tests. https://github.com/BrockU-4P02-Logistics-System/Frontend/tree/mongodb/__tests__

5 User Feedback

For actually getting feedback from the perspective of a user not involved with the development of the system, we got the help of a family member of a group member to provide feedback. The manner in which this was conducted was the family member was sent the link to the system along with the user manual. We provide the feedback on the system in its entirety as it was more detailed than planned. This is available in the documentation repository of our system and is interesting to hear from a potential user.